Balan Paula
Grupa 132

Baze de date
Gestiunea unor cafenele

Studiu de caz

În acestă bază de date are loc gestiunea unor cafenele.

Vom prezenta modelul de date, restricțiile pe care trebuie să le resptecte și contruirea diagramei E/R corespunzătoare.

Modelul de date va gestiona informații legate de organizarea și funcționarea cafenelelor. Cafenelele conțin angajați cu diferite job-uri și se află într-o anumită locație. Cafeneaua conține produse de tip gustări(snacks) sau băuturi(drinks), care sunt împărțite pe categorii. Gustările sunt oferite de un furnizor care se află într-o anumită locație. Băuturile sunt ținute într-un anumit tip de pahar.

Comenzile vor fi eliberate de un angajat, iar o comandă poate conține mai multe tipuri de produse.

Modelul de date respectă anumite restricții de funcționare.

- O cafenea trebuie să aibă minim un angajar
- Un angajat poate lucra la o singură cafenea și are un singur job.
- O cafenea conține minim un produs
- Furnizorii și cafenelele se află într-o locație
- O gustare este achiziționată de la un supplier
- Un supplier poate furniza mai multe gustări
- Produsele pot fi mai multe gustări/băuturi
- Gustările/Băuturile aparțin unei categorii
- Băuturile se află într-un tip de pahar
- Mai multe produse pot aparea în mai multe bonuri
- O comandă conține un produs
- Un bon are mai multe comenzi
- Un bon este eliberat de un angajt care îndeplinește are jobul *cashier*
- Bonurile aparțin unei cafelele
- Cafenelele pot elibera zero sau mai multe bonuri
- Un client trebuie să plătească cel puțin un bon.

Entități

Pentru modelul de date, structurile CAFES, EMPLOYEES, JOBS, LOCATIONS, SUPPLIER, PRODUCTS, SNACKS, DRINKS, CATEGORIES, GLASSES, ORDERS și BILLS reprezintă entități.

Vom prezenta entitățile modelului de date, dând o descriere completă a fiecăreia. De asemenea, pentru fiecare entitate sa va preciza cheia primară.

CAFES = entitate care oferă diverse informații a unei cafenele. Cheia primară a entității este *cafe_id.*

EMPLOYEES = persoană fizică, angajată a unei cafenele ce îndeplinește un animit job. Cheia primară a entității este *employee_id*.

JOBS = identifică rolul unui angajat. Cheia primară a entității este *job_id*.

LOCATIONS = identifică localizarea unei cafenele sau a unui furnizor. Cheia primară a entității este *location_id.*

SUPPLIER = oranizație care distribuie diferite tipuri de gustări. Cheia primară a entității este *supplier_id.*

PRODUCTS = identifică numele și prețul unei gustări sau a unei băuturi din cadrul unei cafenele. Cheia primară a entității este product_id.

SNACKS = produs vândut de o cafenea, achiziționat de la un furnizor. Cheia primară a entității este *snack_id.*

DRINKS = produs vândut de o cafenea. Cheia primară a entității este *drink_id.*

CATEGORIES = identifică categoria din care face parte o anumită gustare sau o anumită băutură. Cheia primară a entității este *category_id.*

GLASSES = entitate dependent de DRINKS, care conține tipul paharului în care este ținută băutura. Cheia primară a entității este *glass_id.*

ORDERS = entitate asociativă care face legătura între *PRODUCTS* și *BILLS*, caere conține cantitatea unui anumit produs dintr-un bon. Cheia primară a entității este *order_id.*

BILLS = entitate care conține informați despre o comandă dintr-o cafenea. Cheia primară a entității este *bill_id.*

CUSTOMERS = persoană fizică, client al unei cafenele ce plătește bonuri. Cheia primară a entității este *customer_id*.

Relații

Vom prezenta relaţiile modelului de date, dând o descriere completă a fiecăreia. De fapt, denumirile acestor legături sunt sugestive, reflectând conţinutul acestora şi entităţile pe care le leagă. Pentru fiecare relaţie se va preciza cardinalitatea minimă şi maxima.

EMPLOYEES lucreaza *CAFES* = relație care leagă entitățile, reflectând legătura dintre acestea (în ce cafenea lucrează un angajat). Ea are cardinalitatea minima *1:1* și cardinalitatea maximă *1:n.*

EMPLOYEES are un *JOBS* = relație care leagă entitățile, reflectând legătura dintre acestea (ce job are un angajat). Ea are cardinalitatea minima *1:0* și cardinalitatea maximă *1:n.*

CAFES are o *LOCATIONS* = relație care leagă entitățile, reflectând legătura dintre acestea (localizarea unei cafenele). Ea are cardinalitatea minima *1:0* și cardinalitatea maximă *1:1*.

SUPPLIER are o *LOCATIONS* = relație care leagă entitățile, reflectând legătura dintre acestea (localizarea furnizatorului). Ea are cardinalitatea minima *1:0* și cardinalitatea maximă *1:1*.

CAFES are *PRODUCTS* = relație care leagă entitățile, reflectând legătura dintre acestea (). Ea are cardinalitatea minima *0:1* și cardinalitatea maximă *1:n.*

PRODUCTS este *SNACKS* = relație care leagă entitățile, reflectând legătura dintre acestea (gustarea este un produs). Ea are cardinalitatea minima *1:0* și cardinalitatea maximă *1:1*.

PRODUCTS este *DRINKS* = relație care leagă entitățile, reflectând legătura dintre acestea (băutura este un produs). Ea are cardinalitatea minima *1:0* și cardinalitatea maximă *1:1*.

SNACKS aparține *CATEGORIES* = relație care leagă entitățile, reflectând legătura dintre acestea (din ce categorie fac parte gustările). Ea are cardinalitatea minima *0:1* și cardinalitatea maximă *n:1*.

DRINKS aparține *CATEGORIES* = relație care leagă entitățile, reflectând legătura dintre acestea (din ce categorie fac parte băuturile). Ea are cardinalitatea minima *0:1* și cardinalitatea maximă *n:1*.

DRINKS se află *GLASSES* = relație care leagă entitățile, reflectând legătura dintre acestea (în ce tip de pahar se află băutura). Ea are cardinalitatea minima *0:1* și cardinalitatea maximă *n:1*.

PRODUCTS orders *BILLS* = relație de tip *many-to-many* care leagă entitățile, reflectând legătura dintre acestea (ce produse aparțin bourilor). Ea are cardinalitatea minima *0:0* și cardinalitatea maximă *m:n.*

ORDERS conține *PRODUCTS* = relație care leagă entitățile, reflectând legătura dintre acestea (comenzile conțin produse). Ea are cardinalitatea minima *1:0* și cardinalitatea maximă *1:m*.

ORDERS aparține *BILLS* = relație care leagă entitățile, reflectând legătura dintre acestea (comenzile aparțin unor bonuri). Ea are cardinalitatea minima *1:0* și cardinalitatea maximă *1:n*.

EMPLOYEES eliberează *BILLS* = relație care leagă entitățile, reflectând legătura dintre acestea (bonurile sunt eliberate de un angajat). Ea are cardinalitatea minima *1:0* și cardinalitatea maximă *1:n*.

Atribute

CAFES are ca atribute:

- cafe_id = variabilă de tip întreg, de lungime maximă de 3, care reprezintă codul cafenelei
- location_id = variabilă de tip întreg, de lungime maximă de 3, care reprezintă codul locației. Nu poate conține informații de tip null.
- cafe_name = variabilă de tip caracter de lungime maximă de 20, care reprezintă numele cafenelei
- cafe_phone_number = variabilă de tip caracter de lungime fixă de 10, care reprezintă numărul de telefon al cafenelei
- cafe_email = variabilă de tip caracter de lungime maximă de 20, care reprezintă adresa cafenelei
- manager_id = variabilă de tip întreg, de lungime maxima de 3, care reprezintă codul managerului cafenelei

EMPLOYEES are ca atribute:

- employees_id = variabilă de tip întreg, de lungime maximă de 3, care reprezintă codul angajatului.
- job_id = variabilă de tip întreg, de lungime maximă de 3, care reprezintă codul unui job. Nu poate conține informații de tip null.
- cafe_id = variabilă de tip întreg, de lungime maximă de 3, care reprezintă codul unei cafenele. Nu poate conține informații de tip null.
- first_name = variabilă de tip caracter, de lungime maximă de 10, care reprezintă numele angajatului.
- last_name = variabilă de tip caracter, de lungime maxima de 10, care reprezintă prenumele angajatului.
- phone_number = variabilă de tip caracter de lungime fixă de 10, care reprezintă numărul de telefon al angajatului.
- email = variabilă de tip caracter de lungime maximă de 20, care reprezintă adresa angajatului.
- salary = variabilă de tip întreg, de lungime maxima de 4, care reprezintă salariul angajatului.
- hire_date = variabilă de tip data calanderistică, care reprezintă data la care angajatul și-a început activitatea.
- manager_id = variabilă de tip întreg, de lungime maxima de 3, care reprezintă codul managerului angajatului (sau null dacă angajatul nu are manager).

JOBS are ca atribute:

- job_id = variabilă de tip întreg, de lungime maxima de 3, care reprezintă codul jobului.
- job_title = variabilă de tip caracter, de lungime maxima de 10, care reprezintă numele jobului.
- min_salary = variabilă de tip întreg, de lungime maxima de 4, care reprezintă salariul minim al jobului.
- max_salary = variabilă de tip întreg, de lungime maxima de 4, care reprezintă salariul maxim al jobului.

LOCATIONS are ca atribute:

- location_id = variabilă de tip întreg, de lungime maxima de 3, care reprezintă codul locației.

- street_address = variabilă de tip caracter, de lungime maxima de 20, care reprezintă adresa locației.
- postal_code = variabilă de tip caracter, de lungime fixă de 6, care reprezintă codul postal.
- city = variabilă de tip caracter, de lungime maximă de 10, care reprezintă numele orașului.

SUPPLIER are ca atribute:

- supplier_id = variabilă de tip întreg, de lungime maximă de 3, care reprezintă codul furnizorului.
- location_id = variabilă de tip întreg, de lungime maximă de 3, care reprezintă codul locației. Nu poate conține informații de tip null.
- supplier_name = variabilă de tip caracter, de lungime maximă de 15, care reprezintă numele furnizorului.
- phone_number = variailă de tip caracter, de lungime fixă de 10, care reprezintă numărul de telefon al furnizorului.
- supplier_email = variabilă de tip caracter, de lungime maxima de 20, care reprezintă emailul furnizorului.

PRODUCTS are ca atribute:

- product_id = variabilă de tip întreg, de lungime maximă de 3, care reprezintă codul produsului.
- cafe_id = variabilă de tip întreg, de lungime maximă de 3, care reprezintă codul cafenelei. Nu poate conține informații de tip null.
- p_name = variabilă de tip caracter, de lungime maximă de 15, care reprezintă numele produsului.
- price = variabilă de tip real, de lungime intreaga maximă de 2 și de lungime fracțională maximă de 2, care reprezintă prețul produsului.

SNACKS are ca atribute:

- snack_id = variabilă de tip întreg, de lungime maximă de 3, care reprezintă codul gustării.
- supplier_id = variabilă de tip întreg, de lungime maximă de 3, care reprezintă codul furnizorului. Nu poate conține informații de tip null.
- product_id = variabilă de tip întreg, de lungime maximă de 3, care reprezintă codul produsului. Nu poate conține informații de tip null.

- category_id = variabilă de tip întreg, de lungime maxima de 3, care reprezintă codul categoriei. Nu poate conține informații de tip null.

DRINKS are ca atribute:

- drink_id = variabilă de tip întreg, de lungime maximă de 3, care reprezintă codul băuturii.
- product_id = variabilă de tip întreg, de lungime maximă de 3, care reprezintă codul produsului. Nu poate conține informații de tip null.
- glass_id = variabilă de tip întreg, de lungime maximă de 3, care reprezintă codul paharului. Nu poate conține informații de tip null.
- category_id = variabilă de tip întreg, de lungime maximă de 3, care reprezintă codul categoriei. Nu poate conține informații de tip null.

CATEGORIES are ca atribute:

- category_id = variabilă de tip întreg, de lungime maximă de 3, care reprezintă codul categoriei.
- category_name = variabilă de tip caracter, de lungime maximă de 10, care reprezintă numele categoriei.

GLASSES are ca atribute:

- glass_id = variabilă de tip întreg, de lungime maximă de 3, care reprezintă codul paharului.
- glass_type = variabilă de tip caracter , de lungime maximă de 10, care reprezintă tipul(numele) paharului.
- glass_material = variabilă de tip caracter, de lungime maximă de 10, care reprezintă materialul din care paharului. De exemplu, poate să fie glass, *porcelain.*

ORDERS are ca atribute:

- order_id = variabilă de tip întreg, de lungime maximă de 3, care reprezintă codul comenzii.
- bill_id = variabilă de tip întreg, de lungime maximă de 3, care reprezintă codul bonului. Nu poate conține informații de tip null.
- product_id = variabilă de tip întreg, de lungime maximă de 3, care reprezintă codul. Nu poate conține informații de tip null.
- quantity = variabilă de tip întreg, de lungime maximă de 1, care reprezintă numărul de produse comandate. Nu poate conține informații de tip null.

BILLS are ca atribute:

- bill_id = variabilă de tip întreg, de lungime maximă de 3, care reprezintă codul bonului.
- employee_id = variabilă de tip întreg, de lungime maximă de 3, care reprezintă codul angajatului. Nu poate conține informații de tip null.
- cafe_id = variabilă de tip întreg, de lungime maximă de 3, care reprezintă codul cafenelei. Nu poate conține informații de tip null.
- customer_id = variabilă de tip întreg, de lungime maximă de 3, care reprezintă codul clientului. Nu poate conține informații de tip null.
- payment_type = variabilă de tip caracter, de lungime fixă de 4, care reprezintă metoda de plată. Valorile trebuie să fie de tipul *cash* sau *card*. Nu poate conține informații de tip null.
- data_bill = variabilă de tip data calendaristică, care reprezintă data la care a fost eliberat bonul. Nu poate conține informații de tip null.

CUSTOMERS are ca atribute:

- customer_id = variabilă de tip întreg, de lungime maximă de 3, care reprezintă codul clientului.
- first_name = variabilă de tip caracter, de lungime maximă de 10, care reprezintă numele clientului.
- last_name = variabilă de tip caracter, de lungime maxima de 10, care reprezintă prenumele clientului.

Scheme relaționale corespunzătoare diagramei conceptuale din figura anterioară sunt următoarele:

CAFES(cafe_id#, location_id, cafe_name, cafe_phone_number, cafe_email, manager_id),
EMPLOYEES(employee_id#, job_id, first_name, last_name, phone_number, email, salary, hire_date, manager_id),
JOBS(job_id#, job_title, min_salary, max_salary),
LOCATIONS(location_id#, street_address, postal_code, city),
SUPPLIERS(supplier_id#, location_id, name, phone_number, supplier_email),
PRODUCTS(product_id#, cafe_id, name, price),
SNACKS(snack_id#, supplier_id, product_id, category_id),
DRINKS(drink_id#, product_id, category_id, glass_id),

CATEGORIES(category_id#, category_name),
GLASSES(glass_id#, glass_type, glass_material),
ORDERS(order_id#, bill_id, product_id, quantity),
BILLS(bill_id#, employee_id, cafe_id, customer_id, payment_type, date),
CUSTOMERS(customer_id#, first_name, last_name).

Normalizări

Schema Non-FN3:

CAFES(cafe_id#, location_id, cafe_name, cafe_phone_number, cafe_email, manager_id),
EMPLOYEES(employee_id#, job_id, first_name, last_name, phone_number, email, salary, hire_date, manager_id),
JOBS(job_id#, job_title, min_salary, max_salary),
LOCATIONS(location_id#, street_address, postal_code, city),
SUPPLIERS(supplier_id#, location_id, name, phone_number, supplier_email),
PRODUCTS(product_id#, cafe_id, name, price),
SNACKS(snack_id#, supplier_id, product_id, category_id),
DRINKS(drink_id#, product_id, category_id, glass_type, glass_material),
CATEGORIES(category_id#, category_name),
ORDERS(order_id#, bill_id, product_id, quantity),
BILLS(bill_id#, employee_id, cafe_id, customer_id, payment_type, date),
CUSTOMERS(customer_id#, first_name, last_name).

De ce nu e FN3: drink_id, glass_type, glass_material (dependent tranzitivă)

Schema Non-FN2:

CAFES(cafe_id#, location_id, cafe_name, cafe_phone_number, cafe_email, manager_id),
EMPLOYEES(employee_id#, job_id, first_name, last_name, phone_number, email, salary, hire_date, manager_id),
JOBS(job_id#, job_title, min_salary, max_salary),
LOCATIONS(location_id#, street_address, postal_code, city),
SUPPLIERS(supplier_id#, location_id, name, phone_number, supplier_email),
PRODUCTS(product_id#, cafe_id, name, price,),
SNACKS(snack_id#, supplier_id, product_id, category_id),

DRINKS(drink_id#, product_id, category_id, glass_type, glass_material),
CATEGORIES(category_id#, category_name),
ORDERS(order_id#, bill_id, product_id, quantity, payment_type),
BILLS(bill_id#, employee_id, cafe_id, customer_id, date),
CUSTOMERS(customer_id#, first_name, last_name).

De ce nu este FN2: payment_type depinde doar de bill_id, nu și de product_id.

Schema Non-FN1:

CAFES(cafe_id#, location_id, cafe_name, cafe_phone_number, cafe_email, manager_id),
EMPLOYEES(employee_id#, job_id, first_name, last_name, phone_number, email, salary, hire_date, manager_id),
JOBS(job_id#, job_title, min_salary, max_salary),
LOCATIONS(location_id#, street_address, postal_code, city),
SUPPLIERS(supplier_id#, location_id, name, phone_number, supplier_email),
PRODUCTS(product_id#, cafe_id, name, price,),
SNACKS(snack_id#, supplier_id, product_id, category_id),
DRINKS(drink_id#, product_id, category_id, glass_type, glass_material),
CATEGORIES(category_id#, category_name),
ORDERS(order_id#, bill_id, product_id, quantity, payment_type),
BILLS(bill_id#, employee_id, cafe_id, customer_id, date, order_id),
CUSTOMERS(customer_id#, first_name, last_name).

De ce nu este FN1: În tabelul ORDERS, coloana order_id nu este indivizibilă/atomică.

Diagrama entitate-relație

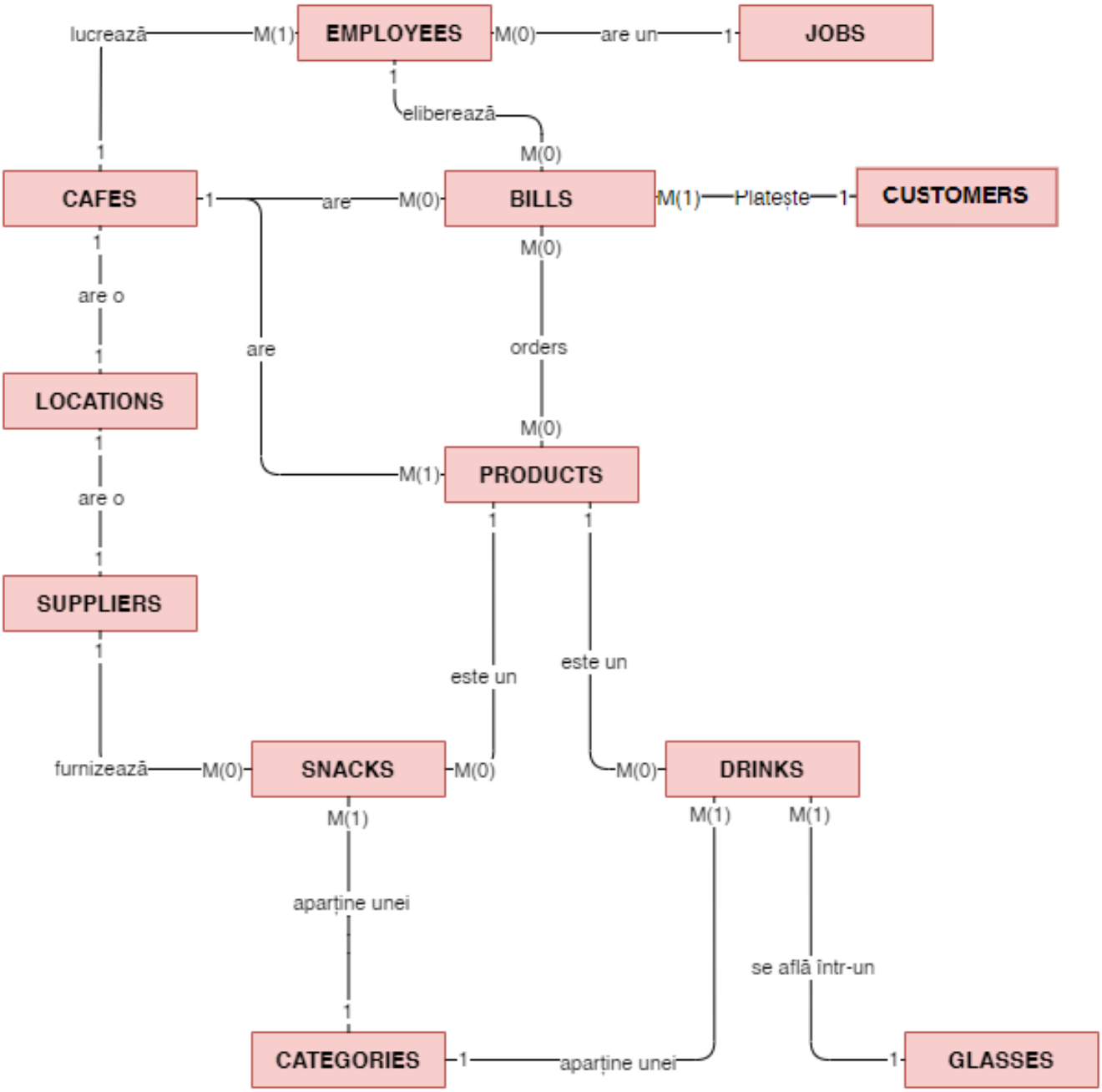
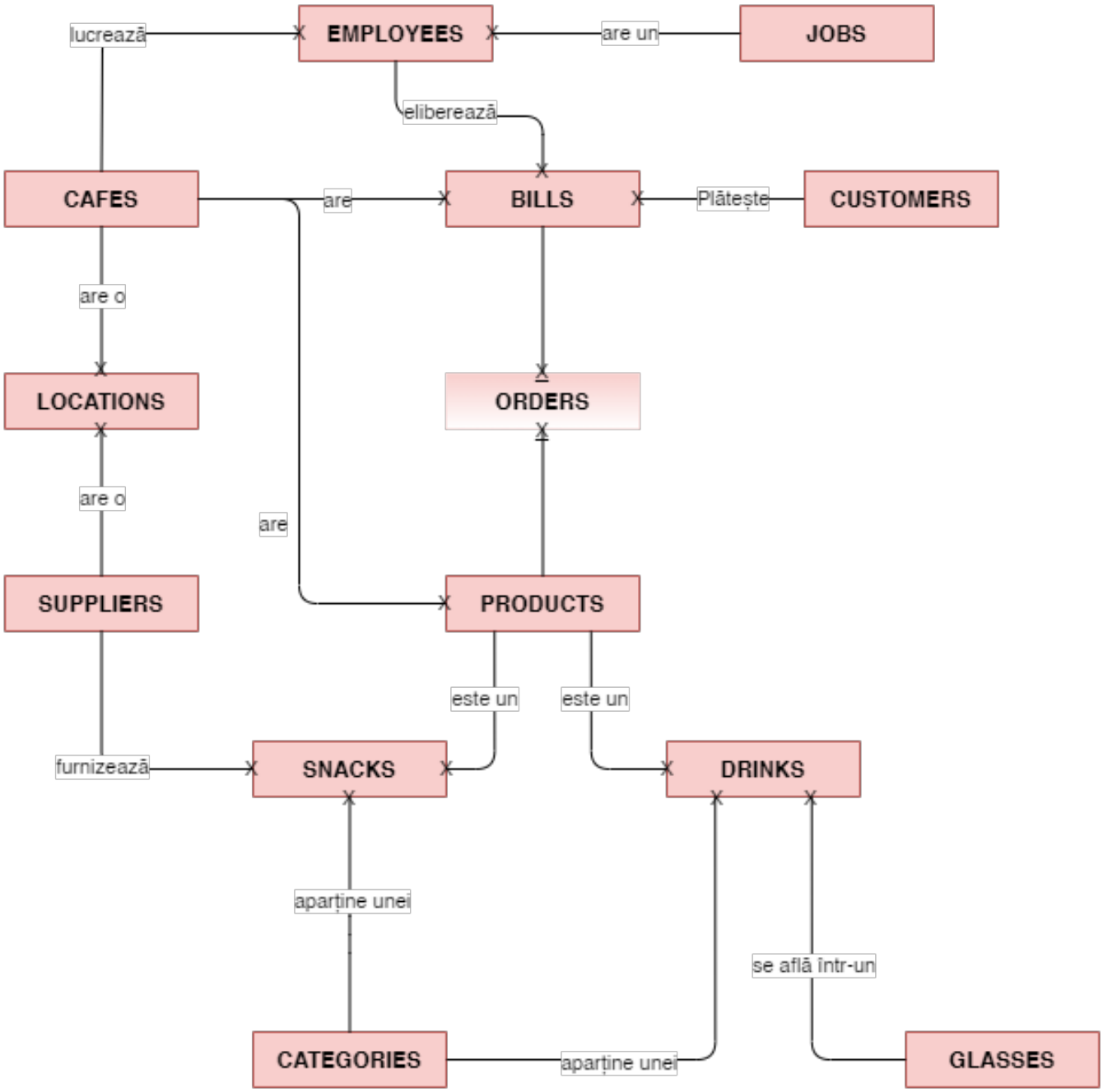


Diagrama conceptuală



Creearea bazei de date

```
CREATE TABLE locations (
    location_id    NUMBER(3) PRIMARY KEY,
    street_address VARCHAR2(20),
    postal_code    CHAR(6),
    city           VARCHAR2(10)
);
```

```
Table LOCATIONS created.
```

```
CREATE TABLE jobs (
    job_id     NUMBER(3) PRIMARY KEY,
    job_title  VARCHAR2(10),
    min_salary NUMBER(4),
    max_salary NUMBER(4)
);
```

```
Table JOBS created.
```

```
CREATE TABLE customers (
    customer_id NUMBER(3) PRIMARY KEY,
    first_name  VARCHAR2(10),
    last_name   VARCHAR2(10)
);
```

```
Table CUSTOMERS created.
```

```
CREATE TABLE categories (
    category_id   NUMBER(3) PRIMARY KEY,
    category_name VARCHAR2(10)
);
```

```
Table CATEGORIES created.
```

```
CREATE TABLE glasses (
    glass_id       NUMBER(3) PRIMARY KEY,
    glass_type     VARCHAR2(10),
    glass_material VARCHAR2(10)
);
```

```
Table GLASSES created.
```

```
CREATE TABLE cafes (
    cafe_id           NUMBER(3) PRIMARY KEY,
    location_id       NUMBER(3) NOT NULL,
    cafe_name         VARCHAR(20),
    cafe_phone_number CHAR(10),
    cafe_email        VARCHAR2(20),
    manager_id        NUMBER(3),
    FOREIGN KEY (location_id) REFERENCES locations(location_id)
);
```

```
Table CAFES created.
```

```
CREATE TABLE suppliers (
    supplier_id NUMBER(3) PRIMARY KEY,
    location_id NUMBER(3) NOT NULL,
    supplier_name VARCHAR2(15),
    phone_number CHAR(10),
    supplier_email VARCHAR(20),
    FOREIGN KEY (location_id) REFERENCES locations(location_id)
);
```

```
Table SUPPLIERS created.
```

```sql
--creare tabel products
CREATE TABLE products (
    product_id NUMBER(3) PRIMARY KEY,
    cafe_id    NUMBER(3) NOT NULL,
    product_name     VARCHAR2(15),
    price      NUMBER(2),
    FOREIGN KEY (cafe_id) REFERENCES cafes(cafe_id)
);
```

```
Table PRODUCTS created.
```

```sql
CREATE TABLE snacks (
    snack_id    NUMBER(3) PRIMARY KEY,
    supplier_id NUMBER(3) NOT NULL,
    product_id  NUMBER(3) NOT NULL,
    category_id NUMBER(3) NOT NULL,
    FOREIGN KEY (supplier_id) REFERENCES suppliers(supplier_id),
    FOREIGN KEY (product_id) REFERENCES products(product_id),
    FOREIGN KEY (category_id) REFERENCES categories(category_id)
);
```

```
Table SNACKS created.
```

```sql
CREATE TABLE drinks (
    drink_id    NUMBER(3) PRIMARY KEY,
    product_id  NUMBER(3) NOT NULL,
    glass_id    NUMBER(3) NOT NULL,
    category_id NUMBER(3) NOT NULL,
    FOREIGN KEY (product_id) REFERENCES products(product_id),
    FOREIGN KEY (glass_id) REFERENCES glasses(glass_id),
    FOREIGN KEY (category_id) REFERENCES categories(category_id)
);
```

```
Table DRINKS created.
```

```sql
CREATE TABLE employees (
    employee_id  NUMBER(3) PRIMARY KEY,
    job_id       NUMBER(3) NOT NULL,
    cafe_id      NUMBER(3) NOT NULL,
    first_name   VARCHAR2(10),
    last_name    VARCHAR2(10),
    phone_number CHAR(10),
    email        VARCHAR2(20),
    salary       NUMBER(4),
    hire_date    DATE,
    manager_id   NUMBER(3),
    FOREIGN KEY (job_id) REFERENCES jobs(job_id),
    FOREIGN KEY (cafe_id) REFERENCES cafes(cafe_id)
);
```

```
Table EMPLOYEES created.
```

```sql
CREATE TABLE bills (
    bill_id      NUMBER(3) PRIMARY KEY,
    employee_id  NUMBER(3) NOT NULL,
    cafe_id      NUMBER(3) NOT NULL,
    customer_id  NUMBER(3) NOT NULL,
    payment_type CHAR(4) NOT NULL,
    data_bill    DATE NOT NULL,
    FOREIGN KEY (customer_id) REFERENCES customers(customer_id)
);
```

```
Table BILLS created.
```

```sql
CREATE TABLE orders (
    order_id   NUMBER(3) PRIMARY KEY,
    bill_id    NUMBER(3) NOT NULL,
    product_id NUMBER(3) NOT NULL,
    quantity   NUMBER(1) NOT NULL,
    FOREIGN KEY (bill_id) REFERENCES bills(bill_id),
    FOREIGN KEY (product_id) REFERENCES products(product_id)
);
```

```
Table ORDERS created.
```

Inserarea datelor

```sql
-- inserarea datelor in tabelul locations
INSERT INTO locations (location_id, street_address, postal_code, city)
VALUES (100, 'Grozavesti 21', '060752', 'Bucuresti');

INSERT INTO locations (location_id, street_address, postal_code, city)
VALUES (101, 'Rovine 15', '110425', 'Pitesti');

INSERT INTO locations (location_id, street_address, postal_code, city)
VALUES (102, 'Traian 20', '100062', 'Ploiesti');

INSERT INTO locations (location_id, street_address, postal_code, city)
VALUES (103, 'Calarasi 52', '030611', 'Bucuresti');

INSERT INTO locations (location_id, street_address, postal_code, city)
VALUES (104, 'Floarea-soarelui 7', '041271', 'Bucuresti');

INSERT INTO locations (location_id, street_address, postal_code, city)
VALUES (105, 'Bucegi 30', '120208', 'Buzau');

INSERT INTO locations (location_id, street_address, postal_code, city)
VALUES (106, 'Carol 20', '030165', 'Bucuresti');

INSERT INTO locations (location_id, street_address, postal_code, city)
VALUES (107, 'Alunului 9', '110335', 'Pitesti');

INSERT INTO locations (location_id, street_address, postal_code, city)
VALUES (108, 'Azuga 1', '100526', 'Ploiesti');

INSERT INTO locations (location_id, street_address, postal_code, city)
VALUES (109, 'Sebes 41', '033084', 'Bucuresti');

INSERT INTO locations (location_id, street_address, postal_code, city)
VALUES (110, 'Gloriei 60', '120057', 'Buzau');

INSERT INTO locations (location_id, street_address, postal_code, city)
VALUES (111, 'Oltului 35', '100361', 'Ploiesti');
```

```sql
-- verificare date
SELECT *
FROM locations;
```

Script Output x | Query Result x

SQL | All Rows Fetched: 13 in 0.003 seconds

	LOCATION_ID	STREET_ADDRESS	POSTAL_CODE	CITY
1	100	Grozavesti 21	060752	Bucuresti
2	101	Rovine 15	110425	Pitesti
3	102	Traian 20	100062	Ploiesti
4	103	Calarasi 52	030611	Bucuresti
5	104	Floarea-soarelui 7	041271	Bucuresti
6	105	Bucegi 30	120208	Buzau
7	106	Carol 20	030165	Bucuresti
8	107	Alunului 9	110335	Pitesti
9	108	Azuga 1	100526	Ploiesti
10	109	Sebes 41	033084	Bucuresti
11	110	Gloriei 60	120057	Buzau
12	111	Oltului 35	100361	Ploiesti
13	112	Parcului 5	012329	Bucuresti

```sql
-- inserarea datelor in tabelul bills
INSERT INTO bills (bill_id, employee_id, cafe_id, payment_type, data_bill)
VALUES (100, 103, 100, 'Cash', to_date('28/Apr/21', 'DD-MON-YY'));

INSERT INTO bills (bill_id, employee_id, cafe_id, payment_type, data_bill)
VALUES (101, 109, 101, 'Card', to_date('20/Apr/21', 'DD-MON-YY'));

INSERT INTO bills (bill_id, employee_id, cafe_id, payment_type, data_bill)
VALUES (102, 113, 100, 'Cash', to_date('5/May/21', 'DD-MON-YY'));

INSERT INTO bills (bill_id, employee_id, cafe_id, payment_type, data_bill)
VALUES (103, 113, 104, 'Card', to_date('6/May/21', 'DD-MON-YY'));

INSERT INTO bills (bill_id, employee_id, cafe_id, payment_type, data_bill)
VALUES (104, 103, 100, 'Card', to_date('30/Mar/21', 'DD-MON-YY'));

INSERT INTO bills (bill_id, employee_id, cafe_id, payment_type, data_bill)
VALUES (105, 117, 101, 'Cash', to_date('31/Mar/21', 'DD-MON-YY'));

INSERT INTO bills (bill_id, employee_id, cafe_id, payment_type, data_bill)
VALUES (106, 124, 103, 'Cash', to_date('15/Apr/21', 'DD-MON-YY'));
```

```sql
SELECT *
FROM BILLS;
```

Script Output | Query Result

SQL | All Rows Fetched: 7 in 0.002 seconds

	BILL_ID	EMPLOYEE_ID	CAFE_ID	PAYMENT_TYPE	DATA_BILL
1	100	103	100	Cash	28-APR-21
2	101	109	101	Card	20-APR-21
3	102	113	100	Cash	05-MAY-21
4	103	113	104	Card	06-MAY-21
5	104	103	100	Card	30-MAR-21
6	105	117	101	Cash	31-MAR-21
7	106	124	103	Cash	15-APR-21

```sql
-- inserarea datelor in tabelul jobs
INSERT INTO jobs (job_id, job_title, min_salary, max_salary)
VALUES (100, 'manager', 2500, 3000);

INSERT INTO jobs (job_id, job_title, min_salary, max_salary)
VALUES (101, 'barista', 2000, 2500);

INSERT INTO jobs (job_id, job_title, min_salary, max_salary)
VALUES (102, 'waiter', 2000, 2500);

INSERT INTO jobs (job_id, job_title, min_salary, max_salary)
VALUES (103, 'cashier', 1500, 2000);

INSERT INTO jobs (job_id, job_title, min_salary, max_salary)
VALUES (104, 'janitor', 1500, 2000);
```

```sql
-- verificare date
SELECT *
FROM jobs;
```

Script Output | Query Result

SQL | All Rows Fetched: 5 in 0.003 seconds

	JOB_ID	JOB_TITLE	MIN_SALARY	MAX_SALARY
1	100	manager	2500	3000
2	101	barista	2000	2500
3	102	waiter	2000	2500
4	103	cashier	1500	2000
5	104	janitor	1500	2000

```sql
-- inserarea datelor in tabelul categories
INSERT INTO categories (category_id, category_name)
VALUES (100, 'Coffee');

INSERT INTO categories (category_id, category_name)
VALUES (101, 'Tea');

INSERT INTO categories (category_id, category_name)
VALUES (102, 'Lemonade');

INSERT INTO categories (category_id, category_name)
VALUES (103, 'Soda');

INSERT INTO categories (category_id, category_name)
VALUES (104, 'Juice');

INSERT INTO categories (category_id, category_name)
VALUES (105, 'Sweets');

INSERT INTO categories (category_id, category_name)
VALUES (106, 'Chips');
```

```sql
-- verificare date
SELECT *
FROM categories;
```

Script Output | Query Result

SQL | All Rows Fetched: 7 in 0.003 seconds

	CATEGORY_ID	CATEGORY_NAME
1	100	Coffee
2	101	Tea
3	102	Lemonade
4	103	Soda
5	104	Juice
6	105	Sweets
7	106	Chips

```sql
-- inserarea datelor in tabelul glasses
INSERT INTO glasses (glass_id, glass_type, glass_material)
VALUES (100, 'teacup', 'porcelain');

INSERT INTO glasses (glass_id, glass_type, glass_material)
VALUES (101, 'bottle', 'glass');

INSERT INTO glasses (glass_id, glass_type, glass_material)
VALUES (102, 'coffeecup', 'porcelain');

INSERT INTO glasses (glass_id, glass_type, glass_material)
VALUES (103, 'farmhouse', 'glass');

INSERT INTO glasses (glass_id, glass_type, glass_material)
VALUES (104, 'delmonico', 'glass');

INSERT INTO glasses (glass_id, glass_type, glass_material)
VALUES (105, 'highball', 'glass');

INSERT INTO glasses (glass_id, glass_type, glass_material)
VALUES (106, 'pint', 'glass');

INSERT INTO glasses (glass_id, glass_type, glass_material)
VALUES (107, 'balloon', 'glass');

INSERT INTO glasses (glass_id, glass_type, glass_material)
VALUES (108, 'nervis', 'porcelain');

INSERT INTO glasses (glass_id, glass_type, glass_material)
VALUES (109, 'glence', 'porcelain');
```

```sql
-- verificare date
SELECT *
FROM glasses;
```

Script Output | Query Result

SQL | All Rows Fetched: 11 in 0.002 seconds

	GLASS_ID	GLASS_TYPE	GLASS_MATERIAL
1	100	teacup	porcelain
2	101	bottle	glass
3	102	coffeecup	porcelain
4	103	farmhouse	glass
5	104	delmonico	glass
6	105	highball	glass
7	106	pint	glass
8	107	balloon	glass
9	108	nervis	porcelain
10	109	glence	porcelain
11	110	lomond	porcelain

```sql
-- inserarea datelor in tabelul cafes
INSERT INTO cafes (cafe_id, location_id, cafe_name, cafe_phone_number, cafe_email, manager_id)
VALUES (100, 101, 'Noir Cafe', '0771846523', 'noircafe', 100);

INSERT INTO cafes (cafe_id, location_id, cafe_name, cafe_phone_number, cafe_email, manager_id)
VALUES (101, 100, 'Boulevard Cafe', '0748453500', 'boulevardcafe1', 105);

INSERT INTO cafes (cafe_id, location_id, cafe_name, cafe_phone_number, cafe_email, manager_id)
VALUES (102, 104, 'Coffe House', '0760035522', 'coffee_house', 106);

INSERT INTO cafes (cafe_id, location_id, cafe_name, cafe_phone_number, cafe_email, manager_id)
VALUES (103, 107, 'Cafe Mania', '0778448526', 'cafemania', 108);

INSERT INTO cafes (cafe_id, location_id, cafe_name, cafe_phone_number, cafe_email, manager_id)
VALUES (104, 111, 'Coffe Planet', '0750845665', 'coffee.planet', 102);

INSERT INTO cafes (cafe_id, location_id, cafe_name, cafe_phone_number, cafe_email, manager_id)
VALUES (105, 112, 'Coffe Point', '0770089564', 'coffee_point', 101);
```

```sql
-- verificare date
SELECT *
FROM cafes;
```

Script Output × | Query Result ×

SQL | All Rows Fetched: 6 in 0.003 seconds

	CAFE_ID	LOCATION_ID	CAFE_NAME	CAFE_PHONE_NUMBER	CAFE_EMAIL	MANAGER_ID
1	100	101	Noir Cafe	0771846523	noircafe	100
2	101	100	Boulevard Cafe	0748453500	boulevardcafe1	105
3	102	104	Coffe House	0760035522	coffee_house	106
4	103	107	Cafe Mania	0778448526	cafemania	108
5	104	111	Coffe Planet	0750845665	coffee.planet	102
6	105	112	Coffe Point	0770089564	coffee_point	101

```sql
-- inserarea datelor in tabelul suppliers
INSERT INTO suppliers (supplier_id, location_id, supplier_name, phone_number, supplier_email)
VALUES (100, 105, 'Alfers', '0763459512', 'alfers24');

INSERT INTO suppliers (supplier_id, location_id, supplier_name, phone_number, supplier_email)
VALUES (101, 110, 'Milka', '0763459972', 'milka.ro');

INSERT INTO suppliers (supplier_id, location_id, supplier_name, phone_number, supplier_email)
VALUES (102, 102, 'Lay''s', '0771548625', 'laysRO');

INSERT INTO suppliers (supplier_id, location_id, supplier_name, phone_number, supplier_email)
VALUES (103, 106, 'Oreo', '0746485243', 'oreo101');

INSERT INTO suppliers (supplier_id, location_id, supplier_name, phone_number, supplier_email)
VALUES (104, 108, 'Lotto', '0761983462', 'lotto_romania');

INSERT INTO suppliers (supplier_id, location_id, supplier_name, phone_number, supplier_email)
VALUES (105, 103, 'Viva', '0741998544', 'viva14');
```

```sql
-- verificare date
SELECT *
FROM suppliers;
```

Script Output × | Query Result ×

SQL | All Rows Fetched: 6 in 0.003 seconds

	SUPPLIER_ID	LOCATION_ID	SUPPLIER_NAME	PHONE_NUMBER	SUPPLIER_EMAIL
1	100	105	Alfers	0763459512	alfers24
2	101	110	Milka	0763459972	milka.ro
3	102	102	Lay's	0771548625	laysRO
4	103	106	Oreo	0746485243	oreo101
5	104	108	Lotto	0761983462	lotto_romania
6	105	103	Viva	0741998544	viva14

```sql
-- inserarea datelor in tabelul drinks
CREATE SEQUENCE sequence_1
START WITH 100
INCREMENT BY 1
MINVALUE 100
MAXVALUE 112
NO CYCLE;

INSERT INTO drinks (drink_id, product_id, category_id, glass_id)
VALUES (sequence_1.nextval, sequence_1.nextval, 100, 102);

INSERT INTO drinks (drink_id, product_id, category_id, glass_id)
VALUES (sequence_1.nextval, sequence_1.nextval, 100, 102);

INSERT INTO drinks (drink_id, product_id, category_id, glass_id)
VALUES (sequence_1.nextval, sequence_1.nextval, 100, 102);

INSERT INTO drinks (drink_id, product_id, category_id, glass_id)
VALUES (sequence_1.nextval, .nextval, 100, 102);

INSERT INTO drinks (drink_id, product_id, category_id, glass_id)
VALUES (sequence_1.nextval, sequence_1.nextval, 101, 100);

INSERT INTO drinks (drink_id, product_id, category_id, glass_id)
VALUES (sequence_1.nextval, sequence_1.nextval, 101, 100);

INSERT INTO drinks (drink_id, product_id, category_id, glass_id)
VALUES (sequence_1.nextval, sequence_1.nextval, 101, 100);

INSERT INTO drinks (drink_id, product_id, category_id, glass_id)
VALUES (sequence_1.nextval, sequence_1.nextval, 103, 101);

INSERT INTO drinks (drink_id, product_id, category_id, glass_id)
VALUES (sequence_1.nextval, sequence_1.nextval, 103, 101);

INSERT INTO drinks (drink_id, product_id, category_id, glass_id)
VALUES (sequence_1.nextval, sequence_1.nextval, 103, 101);

INSERT INTO drinks (drink_id, product_id, category_id, glass_id)
VALUES (sequence_1.nextval, sequence_1.nextval, 102, 109);
```

```sql
-- verificare date
SELECT *
FROM drinks;
```

Script Output × | Query Result ×

SQL | All Rows Fetched: 56 in 0.005 seconds

	DRINK_ID	PRODUCT_ID	GLASS_ID	CATEGORY_ID
1	100	100	102	100
2	101	101	102	100
3	102	102	102	100
4	103	103	102	100
5	104	104	100	101
6	105	105	100	101
7	106	106	100	101
8	107	107	101	103
9	108	108	101	103
10	109	109	101	103
11	110	110	109	102
12	111	111	109	102
13	112	112	108	102
14	113	118	105	104
15	114	119	106	104
16	115	121	102	100
17	116	122	102	100
18	117	123	102	100
19	118	124	102	100
20	119	125	100	101
21	120	126	100	101
22	121	127	101	103
23	122	128	101	103
24	123	132	102	100

```sql
-- inserarea datelor in tabelul snacks
INSERT INTO snacks (snack_id, supplier_id, product_id, category_id)
VALUES (100, 100, 113, 105);

INSERT INTO snacks (snack_id, supplier_id, product_id, category_id)
VALUES (101, 102, 114, 106);

INSERT INTO snacks (snack_id, supplier_id, product_id, category_id)
VALUES (102, 102, 115, 106);

INSERT INTO snacks (snack_id, supplier_id, product_id, category_id)
VALUES (103, 101, 116, 105);

INSERT INTO snacks (snack_id, supplier_id, product_id, category_id)
VALUES (104, 103, 117, 105);

INSERT INTO snacks (snack_id, supplier_id, product_id, category_id)
VALUES (105, 103, 120, 105);

INSERT INTO snacks (snack_id, supplier_id, product_id, category_id)
VALUES (106, 105, 129, 106);

INSERT INTO snacks (snack_id, supplier_id, product_id, category_id)
VALUES (107, 105, 130, 106);

INSERT INTO snacks (snack_id, supplier_id, product_id, category_id)
VALUES (108, 103, 131, 105);

INSERT INTO snacks (snack_id, supplier_id, product_id, category_id)
VALUES (109, 102, 146, 106);

INSERT INTO snacks (snack_id, supplier_id, product_id, category_id)
VALUES (110, 102, 147, 106);

INSERT INTO snacks (snack_id, supplier_id, product_id, category_id)
VALUES (111, 101, 167, 105);
```

```sql
-- verificare date
SELECT *
FROM snacks;
```

Script Output | Query Result

SQL | All Rows Fetched: 13 in 0.002 seconds

	SNACK_ID	SUPPLIER_ID	PRODUCT_ID	CATEGORY_ID
1	100	100	113	105
2	101	102	114	106
3	102	102	115	106
4	103	101	116	105
5	104	103	117	105
6	105	103	120	105
7	106	105	129	106
8	107	105	130	106
9	108	103	131	105
10	109	102	146	106
11	110	102	147	106
12	111	101	167	105
13	112	100	168	105

```oracle
-- inserarea datelor in tabelul employees
INSERT INTO employees (employee_id, job_id, cafe_id, first_name, last_name, phone_number, email, salary, hire_date, manager_id)
VALUES (100, 100, 100, 'Bruce', 'Dilly', '0738217557', 'BDilly', 3000, to_date('28/Apr/21', 'DD-MON-RR'), NULL);

INSERT INTO employees (employee_id, job_id, cafe_id, first_name, last_name, phone_number, email, salary, hire_date, manager_id)
VALUES (101, 101, 100, 'Daniel', 'Fay', '0721887141', 'dfay', 2500, to_date('30/Apr/21', 'DD-MON-RR'), 100);

INSERT INTO employees (employee_id, job_id, cafe_id, first_name, last_name, phone_number, email, salary, hire_date, manager_id)
VALUES (102, 102, 100, 'Luis', 'Marvis', '0724791424', 'lmarvis', 2500, to_date('30/Apr/21', 'DD-MON-RR'), 100);

INSERT INTO employees (employee_id, job_id, cafe_id, first_name, last_name, phone_number, email, salary, hire_date, manager_id)
VALUES (103, 103, 100, 'Shelli', 'Grant', '0784395435', 'SGrant', 2000, to_date('2/May/21', 'DD-MON-RR'), 100);

INSERT INTO employees (employee_id, job_id, cafe_id, first_name, last_name, phone_number, email, salary, hire_date, manager_id)
VALUES (104, 104, 101, 'Michael', 'OConnell', '0739817384', 'MOConnell', 2000, to_date('5/Mar/21', 'DD-MON-RR'), 100);

INSERT INTO employees (employee_id, job_id, cafe_id, first_name, last_name, phone_number, email, salary, hire_date, manager_id)
VALUES (105, 101, 101, 'Jhon', 'Russell', '0767500120', 'JRussell', 2400, to_date('5/Mar/21', 'DD-MON-RR'), 100);

INSERT INTO employees (employee_id, job_id, cafe_id, first_name, last_name, phone_number, email, salary, hire_date, manager_id)
VALUES (106, 101, 101, 'Trenna', 'Matos', '0767589667', 'TMatos', 2450, to_date('10/May/21', 'DD-MON-RR'), 100);

INSERT INTO employees (employee_id, job_id, cafe_id, first_name, last_name, phone_number, email, salary, hire_date, manager_id)
VALUES (107, 102, 101, 'Laura', 'Olsen', '0793310740', 'lolsen', 2300, to_date('8/May/21', 'DD-MON-RR'), 100);

INSERT INTO employees (employee_id, job_id, cafe_id, first_name, last_name, phone_number, email, salary, hire_date, manager_id)
VALUES (108, 104, 101, 'Sophie', 'Sully', '0711237299', 'ssully', 1700, to_date('5/Apr/21', 'DD-MON-RR'), 100);

INSERT INTO employees (employee_id, job_id, cafe_id, first_name, last_name, phone_number, email, salary, hire_date, manager_id)
VALUES (109, 103, 101, 'Peter', 'Smith', '0746433608', 'PSmith', 1900, to_date('7/Mar/21', 'DD-MON-RR'), 100);
```

```oracle
-- verificare date
SELECT *
FROM employees;
```

Script Output × | Query Result ×

SQL | All Rows Fetched: 25 in 0.002 seconds

	EMPLOYEE_ID	JOB_ID	CAFE_ID	FIRST_NAME	LAST_NAME	PHONE_NUMBER	EMAIL	SALARY	HIRE_DATE	MANAGER_ID
1	100	100	100	Bruce	Dilly	0738217557	BDilly	3000	28-APR-21	(null)
2	101	101	100	Daniel	Fay	0721887141	dfay	2500	30-APR-21	100
3	102	102	100	Luis	Marvis	0724791424	lmarvis	2500	30-APR-21	100
4	103	103	100	Shelli	Grant	0784395435	SGrant	2000	02-MAY-21	100
5	104	104	101	Michael	OConnell	0739817384	MOConnell	2000	05-MAR-21	100
6	105	101	101	Jhon	Russell	0767500120	JRussell	2400	05-MAR-21	100
7	106	101	101	Trenna	Matos	0767589667	TMatos	2450	10-MAY-21	100
8	107	102	101	Laura	Olsen	0793310740	lolsen	2300	08-MAY-21	100
9	108	104	101	Sophie	Sully	0711237299	ssully	1700	05-APR-21	100
10	109	103	101	Peter	Smith	0746433608	PSmith	1900	07-MAR-21	100
11	110	100	102	Clara	Ozer	0772806573	cozer	2500	22-MAY-21	(null)
12	111	100	103	Sarath	Bloom	0714926559	sbloom	2570	10-MAR-21	(null)
13	112	101	103	Allan	Fox	0763800455	afox	2000	18-MAR-21	111
14	113	103	103	Danielle	Russell	0787965155	drussell	1850	12-MAR-21	111
15	114	100	104	Jack	Abel	0797005558	JAbel	2800	31-MAR-21	(null)
16	115	101	104	Julia	Roberts	0735753702	jroberts	2500	15-APR-21	114
17	116	102	104	Alexis	Hall	0738478164	ahall	2000	17-APR-21	114
18	117	103	104	Martha	Vargas	0765007077	mvargas	1500	30-APR-21	114
19	118	104	104	Charles	Gee	0762910685	cgee	1500	07-MAY-21	114
20	119	101	105	Ellen	Landry	0781997387	ELandry	2400	19-MAY-21	114
21	120	102	105	Tayler	Red	0711624052	tred	2300	19-MAY-21	114
22	121	100	105	Felix	Fay	0755671789	felixfay	2750	22-APR-21	(null)
23	122	101	105	Susan	Russell	0759546319	srussell	2200	25-APR-21	121
24	123	102	105	Michael	Lee	0726437768	mlee	2250	18-MAY-21	121
25	124	103	105	Allan	Bull	0761548159	ABULL	2050	20-MAY-21	121

```sql
-- inserarea datelor in tabelul customers
INSERT INTO customers (customer_id, first_name, last_name)
VALUES (100, 'Jenkins', 'Oliver');

INSERT INTO customers (customer_id, first_name, last_name)
VALUES (101, 'Arhold', 'Rebecca');

INSERT INTO customers (customer_id, first_name, last_name)
VALUES (102, 'White', 'Milly');

INSERT INTO customers (customer_id, first_name, last_name)
VALUES (103, 'Rivera', 'Ross');

INSERT INTO customers (customer_id, first_name, last_name)
VALUES (104, 'Brown', 'Jay');
```

```sql
-- verificare date
SELECT *
FROM customers;
```

Script Output × | Query Result ×

SQL | All Rows Fetched: 5 in 0.002 seconds

	CUSTOMER_ID	FIRST_NAME	LAST_NAME
1	100	Jenkins	Oliver
2	101	Arhold	Rebecca
3	102	White	Milly
4	103	Rivera	Ross
5	104	Brown	Jay

```sql
-- inserarea datelor in tabelul bills
INSERT INTO bills (bill_id, employee_id, cafe_id, customer_id, payment_type, data_bill)
VALUES (100, 103, 100, 100, 'Cash', to_date('15/May/21', 'DD-MON-RR'));

INSERT INTO bills (bill_id, employee_id, cafe_id, customer_id, payment_type, data_bill)
VALUES (101, 109, 101, 101, 'Card', to_date('20/Apr/21', 'DD-MON-RR'));

INSERT INTO bills (bill_id, employee_id, cafe_id, customer_id, payment_type, data_bill)
VALUES (102, 113, 100, 101, 'Cash', to_date('5/May/21', 'DD-MON-RR'));

INSERT INTO bills (bill_id, employee_id, cafe_id, customer_id, payment_type, data_bill)
VALUES (103, 113, 104, 102, 'Card', to_date('6/May/21', 'DD-MON-RR'));

INSERT INTO bills (bill_id, employee_id, cafe_id, customer_id, payment_type, data_bill)
VALUES (104, 103, 100, 102, 'Card', to_date('16/Mar/21', 'DD-MON-RR'));

INSERT INTO bills (bill_id, employee_id, cafe_id, customer_id, payment_type, data_bill)
VALUES (105, 117, 101, 103, 'Cash', to_date('30/Apr/21', 'DD-MON-RR'));

INSERT INTO bills (bill_id, employee_id, cafe_id, customer_id, payment_type, data_bill)
VALUES (106, 124, 103, 104, 'Cash', to_date('15/Mar/21', 'DD-MON-RR'));
```

```sql
-- verificare date
SELECT *
FROM bills;
```

Script Output × | Query Result ×

SQL | All Rows Fetched: 7 in 0.005 seconds

	BILL_ID	EMPLOYEE_ID	CAFE_ID	CUSTOMER_ID	PAYMENT_TYPE	DATA_BILL
1	100	103	100	100	Cash	15-MAY-21
2	101	109	101	101	Card	20-APR-21
3	102	113	100	101	Cash	05-MAY-21
4	103	113	104	102	Card	06-MAY-21
5	104	103	100	102	Card	16-MAR-21
6	105	117	101	103	Cash	30-APR-21
7	106	124	103	104	Cash	15-MAR-21

```sql
-- inserarea datelor in tabelul products
INSERT INTO products (product_id, cafe_id, product_name, price)
VALUES (100, 101, 'espresso', 5);

INSERT INTO products (product_id, cafe_id, product_name, price)
VALUES (101, 101, 'americano', 5);

INSERT INTO products (product_id, cafe_id, product_name, price)
VALUES (102, 101, 'cappuccino', 8);

INSERT INTO products (product_id, cafe_id, product_name, price)
VALUES (103, 101, 'caffe latte', 7);

INSERT INTO products (product_id, cafe_id, product_name, price)
VALUES (104, 101, 'green tea', 7);

INSERT INTO products (product_id, cafe_id, product_name, price)
VALUES (105, 101, 'black tea', 7);

INSERT INTO products (product_id, cafe_id, product_name, price)
VALUES (106, 101, 'fruits tea', 7);

INSERT INTO products (product_id, cafe_id, product_name, price)
VALUES (107, 101, 'cola', 8);

INSERT INTO products (product_id, cafe_id, product_name, price)
VALUES (108, 101, 'fanta', 8);

INSERT INTO products (product_id, cafe_id, product_name, price)
VALUES (109, 101, 'sprite', 8);

INSERT INTO products (product_id, cafe_id, product_name, price)
VALUES (110, 101, 'lemonade', 10);
```

```sql
-- verificare date
SELECT *
FROM products;
```

Script Output | Query Result

SQL | All Rows Fetched: 69 in 0.004 seconds

	PRODUCT...	CAFE_ID	PRODUCT_NAME	PRICE
1	100	101	espresso	5
2	101	101	americano	5
3	102	101	cappuccino	8
4	103	101	caffe latte	7
5	104	101	green tea	7
6	105	101	black tea	7
7	106	101	fruits tea	7
8	107	101	cola	8
9	108	101	fanta	8
10	109	101	sprite	8
11	110	101	lemonade	10
12	111	101	mint lemonade	12
13	112	101	peach lemonade	12
14	113	101	wafers	4
15	114	101	salt chips	5
16	115	101	peper chips	5
17	116	101	milka choco...	5
18	117	101	oreo brownies	6
19	118	101	apple juice	10
20	119	101	orange juice	10

```sql
-- inserarea datelor in tabelul orders
INSERT INTO orders (order_id, bill_id, product_id, quantity)
VALUES (100, 100, 143, 5);

INSERT INTO orders (order_id, bill_id, product_id, quantity)
VALUES (101, 106, 135, 7);

INSERT INTO orders (order_id, bill_id, product_id, quantity)
VALUES (102, 105, 101, 2);

INSERT INTO orders (order_id, bill_id, product_id, quantity)
VALUES (103, 102, 130, 3);

INSERT INTO orders (order_id, bill_id, product_id, quantity)
VALUES (104, 103, 153, 1);

INSERT INTO orders (order_id, bill_id, product_id, quantity)
VALUES (105, 104, 131, 5);

INSERT INTO orders (order_id, bill_id, product_id, quantity)
VALUES (106, 101, 105, 4);

INSERT INTO orders (order_id, bill_id, product_id, quantity)
VALUES (107, 100, 128, 6);

INSERT INTO orders (order_id, bill_id, product_id, quantity)
VALUES (108, 101, 101, 3);

INSERT INTO orders (order_id, bill_id, product_id, quantity)
VALUES (109, 101, 112, 8);

INSERT INTO orders (order_id, bill_id, product_id, quantity)
VALUES (110, 105, 106, 2);

INSERT INTO orders (order_id, bill_id, product_id, quantity)
VALUES (111, 105, 120, 2);

INSERT INTO orders (order_id, bill_id, product_id, quantity)
VALUES (112, 102, 129, 6);

INSERT INTO orders (order_id, bill_id, product_id, quantity)
VALUES (113, 103, 148, 5);
```

```sql
-- verificare date
SELECT *
FROM orders;
```

Script Output | Query Result

SQL | All Rows Fetched: 14 in 0.003 seconds

	ORDER_ID	BILL_ID	PRODUCT_ID	QUANTITY
1	100	100	143	5
2	101	106	135	7
3	102	105	101	2
4	103	102	130	3
5	104	103	153	1
6	105	104	131	5
7	106	101	105	4
8	107	100	128	6
9	108	101	101	3
10	109	101	112	8
11	110	105	106	2
12	111	105	120	2
13	112	102	129	6
14	113	103	148	5

Rulare cereri

Exercitiul 11

```
--1. Sa se afiseze numarul comenzii, cantitatea, numele cafenelei, pretul comenzii, numele categoriei, si numele paharului
--bauturilor comandate din toate cafenelele, sortate dupa numele cafenelelor, numele bauturilor.
SELECT order_id, quantity, cafe_name, p.price * o.quantity "PRET", category_name, glass_type, glass_material
FROM orders o
INNER JOIN products p
        ON p.product_id = o.product_id
INNER JOIN cafes cf
        ON cf.cafe_id = p.cafe_id
INNER JOIN drinks d
        ON p.product_id = d.product_id
LEFT JOIN  categories c
       ON  d.category_id = c.category_id
LEFT JOIN  glasses g
       ON  d.glass_id = g.glass_id
ORDER by cafe_name, p.product_name;
```

Script Output × Query Result ×

SQL | All Rows Fetched: 10 in 0.003 seconds

	ORDER_ID	QUANTITY	CAFE_NAME	PRET	CATEGORY_NAME	GLASS_TYPE	GLASS_MATERIAL
1	108	3	Boulevard Cafe	15	Coffee	coffeecup	porcelain
2	102	2	Boulevard Cafe	10	Coffee	coffeecup	porcelain
3	106	4	Boulevard Cafe	28	Tea	teacup	porcelain
4	110	2	Boulevard Cafe	14	Tea	teacup	porcelain
5	109	8	Boulevard Cafe	96	Lemonade	nervis	porcelain
6	100	5	Cafe Mania	30	Juice	pint	glass
7	101	7	Coffe House	42	Coffee	coffeecup	porcelain
8	104	1	Coffe Planet	7	Tea	teacup	porcelain
9	113	5	Coffe Planet	25	Coffee	coffeecup	porcelain
10	107	6	Noir Cafe	42	Soda	bottle	glass

```
--2.Sa se afişeze numele, data angajarii, salariul si o coloana reprezentand salariul dupa ce se aplica o marire, astfel:
--pentru salariaţii angajaţi in marite cresterea este de 20%, pentru cei angajati în aprilie cresterea este de 10%, iar
--pentru salariatii angajati dupa luna mai nu se modifica, pentru toti angajatii care au salariul initial mai mare de 2000.
-- 2  rezolvari DECODE si CASE

--a)Rezolvarea DECODE
SELECT last_name || ' ' || first_name "Nume", hire_date, salary "Salariu initial",
       salary * DECODE(to_char(hire_date, 'MM'), 3, 1.2, 4, 1.1, 1) "Salariu_marit_DECODE"
FROM employees
where salary > 2000
ORDER BY 2;
```

Script Output × Query Result ×

SQL | All Rows Fetched: 16 in 0.002 seconds

	Nume	HIRE_DATE	Salariu initial	Salariu_marit_DECODE
1	Russell Jhon	05-MAR-21	2400	2880
2	Bloom Sarath	10-MAR-21	2570	3084
3	Abel Jack	31-MAR-21	2800	3360
4	Roberts Julia	15-APR-21	2500	2750
5	Fay Felix	22-APR-21	2750	3025
6	Russell Susan	25-APR-21	2200	2420
7	Dilly Bruce	28-APR-21	3000	3300
8	Marvis Luis	30-APR-21	2500	2750
9	Fay Daniel	30-APR-21	2500	2750
10	Olsen Laura	08-MAY-21	2300	2300
11	Matos Trenna	10-MAY-21	2450	2450
12	Lee Michael	18-MAY-21	2250	2250
13	Landry Ellen	19-MAY-21	2400	2400
14	Red Tayler	19-MAY-21	2300	2300
15	Bull Allan	20-MAY-21	2050	2050
16	Ozer Clara	22-MAY-21	2500	2500

```
--b)Rezolvarea CASE
SELECT last_name || ' ' || first_name "Nume", hire_date, salary "Salariu initial",
       CASE WHEN to_char(hire_date, 'MM') = 3 THEN salary * 1.2
            WHEN to_char(hire_date, 'MM') = 4 THEN salary * 1.1
            ELSE salary
       END "Salariu_marit_CASE"
FROM employees
where salary > 2000
ORDER BY 2;
```

Script Output | Query Result

SQL | All Rows Fetched: 16 in 0.003 seconds

	Nume	HIRE_DATE	Salariu initial	Salariu_marit_CASE
1	Russell Jhon	05-MAR-21	2400	2880
2	Bloom Sarath	10-MAR-21	2570	3084
3	Abel Jack	31-MAR-21	2800	3360
4	Roberts Julia	15-APR-21	2500	2750
5	Fay Felix	22-APR-21	2750	3025
6	Russell Susan	25-APR-21	2200	2420
7	Dilly Bruce	28-APR-21	3000	3300
8	Marvis Luis	30-APR-21	2500	2750
9	Fay Daniel	30-APR-21	2500	2750
10	Olsen Laura	08-MAY-21	2300	2300
11	Matos Trenna	10-MAY-21	2450	2450
12	Lee Michael	18-MAY-21	2250	2250
13	Landry Ellen	19-MAY-21	2400	2400
14	Red Tayler	19-MAY-21	2300	2300
15	Bull Allan	20-MAY-21	2050	2050
16	Ozer Clara	22-MAY-21	2500	2500

```
--3.Sa se afiseze informatii despre salariatii care castiga mai bine decat cea mai mica medie reala a salariilor pe job-uri in
-- fromatul urmator: "Angajatul <name> este condus de {<manager_id> | nimeni}.
SELECT last_name || ' ' || first_name || ' este condus de ' || nvl(to_char(manager_id), 'nimeni')  "Informatii angajat"
FROM employees
WHERE salary > (SELECT MIn(emp. media_reala)
                FROM (SELECT job_id, AVG(salary) media_reala
                      FROM (SELECT job_id, salary
                            FROM employees)
                      GROUP BY job_id) emp);
```

Script Output | Query Result

SQL | All Rows Fetched: 22 in 0.003 seconds

	Informatii angajat
1	Dilly Bruce este condus de nimeni
2	Fay Daniel este condus de 100
3	Marvis Luis este condus de 100
4	Grant Shelli este condus de 100
5	OConnell Michael este condus de 100
6	Russell Jhon este condus de 100
7	Matos Trenna este condus de 100
8	Olsen Laura este condus de 100
9	Smith Peter este condus de 100
10	Ozer Clara este condus de nimeni
11	Bloom Sarath este condus de nimeni
12	Fox Allan este condus de 111
13	Russell Danielle este condus de 111
14	Abel Jack este condus de nimeni
15	Roberts Julia este condus de 114
16	Hall Alexis este condus de 114
17	Landry Ellen este condus de 114
18	Red Tayler este condus de 114
19	Fay Felix este condus de nimeni
20	Russell Susan este condus de 121
21	Lee Michael este condus de 121
22	Bull Allan este condus de 121

```sql
--4.Sa se afiseze angajatii care au salariul mai mare decat media tututor angajatilor din data de baza si a caror prenume
--se termina intr-o vocala.
WITH emp(medie_sal) as
     (SELECT AVG(salary)
      FROM employees)
SELECT last_name || ' ' || first_name "Nume angajat", salary
FROM employees, emp
WHERE salary > medie_sal
  AND (LOWER(first_name) LIKE '%a'
    OR LOWER(first_name) LIKE '%e'
    OR LOWER(first_name) LIKE '%i'
    OR LOWER(first_name) LIKE '%o'
    OR LOWER(first_name) LIKE '%u');
```

Script Output | Query Result

SQL | All Rows Fetched: 5 in 0.002 seconds

	Nume angajat	SALARY
1	Dilly Bruce	3000
2	Matos Trenna	2450
3	Olsen Laura	2300
4	Ozer Clara	2500
5	Roberts Julia	2500

```sql
--5.Sa se afiseze comenzile care se afla in pahare tip 'teacup' sau 'coffecup'
SELECT *
FROM orders
WHERE product_id IN (SELECT product_id FROM products
                     WHERE product_id IN (SELECT product_id FROM drinks
                                          WHERE glass_id IN (SELECT glass_id FROM glasses
                                                             WHERE UPPER(glass_type) LIKE 'TEACUP'
                                                                OR UPPER(glass_type) LIKE 'COFFEECUP')));
```

Script Output | Query Result

SQL | All Rows Fetched: 7 in 0.005 seconds

	ORDER_ID	BILL_ID	PRODUCT_ID	QUANTITY
1	108	101	101	3
2	102	105	101	2
3	106	101	105	4
4	110	105	106	2
5	101	106	135	7
6	113	103	148	5
7	104	103	153	1

```sql
--6.Sa se afiseze toti salariatii care au fost angajati luna trecuta.
SELECT *
FROM employees
WHERE to_char(hire_date, 'MM') LIKE (SELECT to_char(add_months(sysdate, -1), 'MM')
                                     FROM dual);
```

Script Output | Query Result

SQL | All Rows Fetched: 9 in 0.002 seconds

	EMPLOYEE_ID	JOB_ID	CAFE_ID	FIRST_NAME	LAST_NAME	PHONE_NUMBER	EMAIL	SALARY	HIRE_DATE	MANAGER_ID
1	100	100	100	Bruce	Dilly	0738217557	BDilly	3000	28-APR-21	(null)
2	101	101	100	Daniel	Fay	0721887141	dfay	2500	30-APR-21	100
3	102	102	100	Luis	Marvis	0724791424	lmarvis	2500	30-APR-21	100
4	108	104	101	Sophie	Sully	0711237299	ssully	1700	05-APR-21	100
5	115	101	104	Julia	Roberts	0735753702	jroberts	2500	15-APR-21	114
6	116	102	104	Alexis	Hall	0738478164	ahall	2000	17-APR-21	114
7	117	103	104	Martha	Vargas	0765007077	mvargas	1500	30-APR-21	114
8	121	100	105	Felix	Fay	0755671789	felixfay	2750	22-APR-21	(null)
9	122	101	105	Susan	Russell	0759546319	srussell	2200	25-APR-21	121

```
--7.Sa se afiseze numarul cafenelelor din fiecare oras sub forma "In orasul <city> se afla count<location_id> cafenele."
SELECT 'In orasul ' || city || ' se afla ' || COUNT(location_id) || ' cafenele.' "Informatii"
FROM locations
WHERE location_id IN (SELECT location_id FROM cafes)
GROUP BY city
ORDER BY city;
```

Script Output × Query Result ×

SQL | All Rows Fetched: 3 in 0.002 seconds

	Informatii
1	In orasul Bucuresti se afla 3 cafenele.
2	In orasul Pitesti se afla 2 cafenele.
3	In orasul Ploiesti se afla 1 cafenele.

Exercitiul 12

```
--1.Sa se mareasca cu 10% salariile angajatilor din cafeneaua cu id 101.
UPDATE employees
SET salary = salary * 1.1
WHERE cafe_id = 101;
```

```
6 rows updated.
```

```
--2.Sa se actualizeze salariul maxim din tabelul jobs.
UPDATE jobs
SET max_salary = (SELECT MAX(salary)
                  FROM employees e
                  WHERE jobs.job_id = e.job_id);
```

I

```
5 rows updated.
```

```
--3.Sa se adauge o noua coloana 'amount' in tabelul bills care sa contina suma totala a bonului.
ALTER TABLE bills
ADD amount NUMBER(3);

UPDATE bills
SET amount = (SELECT SUM(p.price * o.quantity) "amount"
              FROM orders o
              INNER JOIN products p
                      ON p.product_id = o.product_id
              WHERE bills.bill_id = o.bill_id
              GROUP BY o.bill_id);
```

```
Table BILLS altered.


7 rows updated.
```

```
--verificare
SELECT *
FROM bills;
```

Script Output × Query Result ×

SQL | All Rows Fetched: 7 in 0.002 seconds

	BILL_ID	EMPLOYEE_ID	CAFE_ID	CUSTOMER_ID	PAYMENT_TYPE	DATA_BILL	AMOUNT
1	100	103	100	100	Cash	15-MAY-21	72
2	101	109	101	101	Card	20-APR-21	139
3	102	113	100	101	Cash	05-MAY-21	45
4	103	113	104	102	Card	06-MAY-21	32
5	104	103	100	102	Card	16-MAR-21	30
6	105	117	101	103	Cash	30-APR-21	36
7	106	124	103	104	Cash	15-MAR-21	42

```
--4.Sa se moifice data de la bonul 100 cu data curenta
UPDATE bills
SET data_bill = (SELECT sysdate
                 FROM dual)
WHERE bill_id = 100;
```

```
1 row updated.
```

Exercitiu 17

```
---17.Afisati numele clientilor care au cumparat cel putin un americano in cel mai scump bon al clientului.
SELECT last_name || ' ' || first_name "Nume clienti"
FROM customers c
JOIN bills b
  ON b.customer_id = c.customer_id
JOIN orders o
  ON b.bill_id = o.bill_id
JOIN products p
  ON p.product_id = o.product_id
WHERE LOWER(product_name) LIKE 'americano'
  AND b.amount = (SELECT MAX(amount)
                  FROM bills b1
                  WHERE b1.customer_id = c.customer_id);
```

Script Output | Query Result

SQL | All Rows Fetched: 2 in 0.007 seconds

	Nume clienti
1	Ross Rivera
2	Rebecca Arhold

Cererea în algebrică relațională:

R1 = PROJECT (CUSTOMERS, last_name || ‘ ’ || first_name, customer_id)
S1 = SELECT(PRODUCTS, product_name = ‘americano’)
S2 = PROJECT(S1, product_id)
T1 = PROJECT(BILLS, amount, bill_id, customer_id)
T2 = SELECT(T1, amount = max(amount))
J1 = JOIN(R1, T2)
J2 = JOIN(J1, ORDERS)
Rezultat = JOIN(J2, S2)

Expresia algebrică

$R_1 = \Pi_{first_name \,||\, ' ' \,||\, last_name,\ customer_id}(CUSTOMERS)$

$S_1 = \sigma_{product_name = 'americano'}(PRODUCTS)$

$S_2 = \Pi_{product_id}(S_1)$

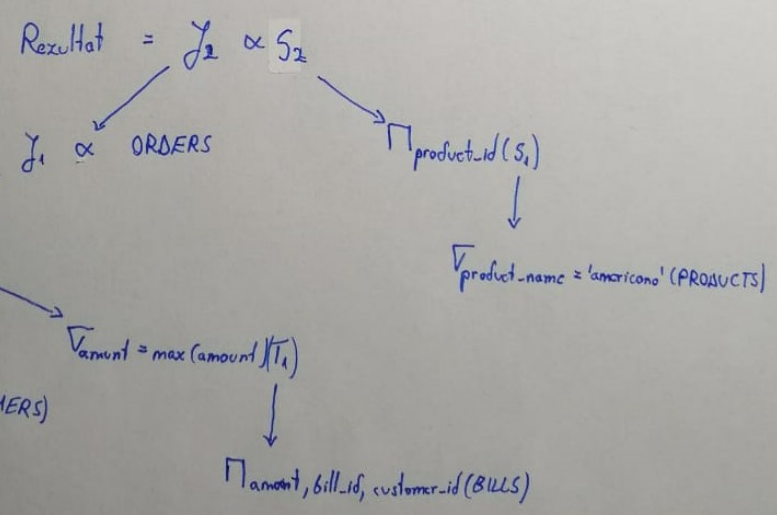
$T_1 = \Pi_{amount,\ bill_id,\ customer_id}(BILLS)$

$T_2 = \sigma_{amount = max(amount)}(T_1)$

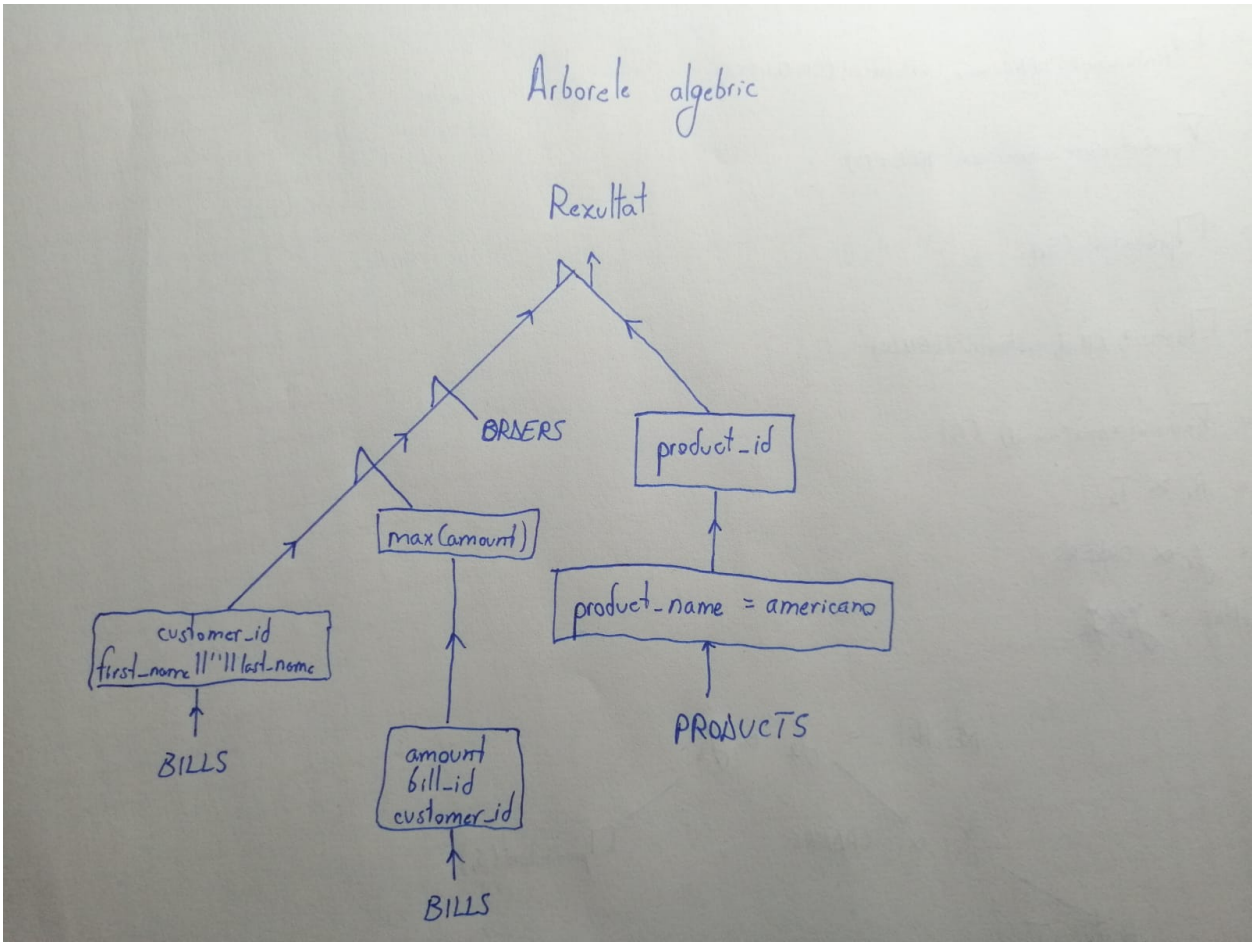
$J_1 = R_1 \propto T_2$

$J_2 = J_1 \propto ORDERS$

$Rezultat = J_2 \propto S_2$



$(\Pi_{first_name \,||\, ' ' \,||\, last_name}(CUSTOMERS) \propto (\sigma_{amount = max(amount)}(\Pi_{amount,\ bill_id,\ customer_id}(BILLS))) \propto ORDERS) \propto (\Pi_{product_id}(\sigma_{product_name = 'americano'}(PRODUCTS)$


Arborele algebric
Rezultat
BRAERS
product_id
max(amount)
product_name = americano
customer_id
first_name || ' ' || last_name
BILLS
amount
bill_id
customer_id
BILLS
PRODUCTS